Intro to Deep Learning

Adrien Aguila--Multner

ENSEIRB-MATMECA
3A CISD

October 13, 2024

Data

- **Training Data**: Consists of input-output pairs (X, Y).
- **Features**: The input variables X used to make predictions.
- **Labels**: The output variables Y that the model aims to predict.
- **Dataset Splitting**:
    - **Training Set**: Used to train the model.
    - **Validation Set**: Used to tune hyperparameters (optional).
    - **Test Set**: Used to evaluate the model's performance.

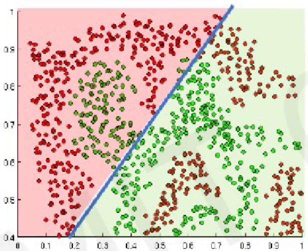
Multilayer Perceptron (MLP)

- **Linear layer**:

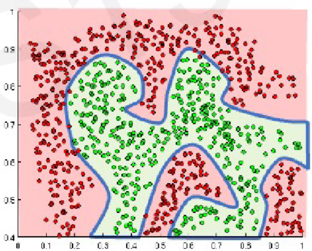
$$\mathbf{y} = \mathbf{W}\mathbf{x} + \mathbf{b} \tag{1}$$

- **Activation Functions**:
  - **ReLU** (Rectified Linear Unit):

$$\text{ReLU}(\mathbf{x}) = \max(0, \mathbf{x}) \tag{2}$$



Linear activation functions produce linear decisions no matter the network size



Non-linearities allow us to approximate arbitrarily complex functions

Loss Function

- **Purpose**: Measures the distance between predicted outputs and true labels.
- **Common Loss Functions**:
  - **Mean Squared Error (MSE)**:

    $$\text{MSE} = \frac{1}{n}\sum_{i=1}^{n}(y_i - \hat{y}_i)^2$$

  - **Cross-Entropy Loss**:

    $$\text{Cross-Entropy} = -\frac{1}{n}\sum_{i=1}^{n}\left(y_i \log(\hat{y}_i) + (1 - y_i)\log(1 - \hat{y}_i)\right)$$

- **Role in Training**: Guides the optimization process to minimize prediction errors.

Gradient Descent

An optimization algorithm used to minimize the loss function iteratively.

Update Rule:

$$\theta := \theta - \eta \nabla_\theta \mathcal{L} \tag{3}$$

where:

- θ are the model parameters.
- η is the learning rate.
- $\nabla_\theta \mathcal{L}$ is the gradient of the loss with respect to θ.

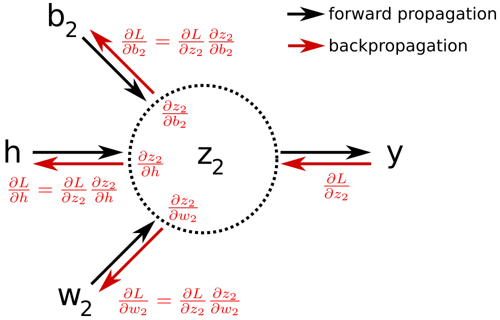
Variants: Adam, RMSprop, etc.

Backpropagation

How to optimize parameters ?

Process:

1. **Forward Pass**: Compute the output and loss
2. **Backward Pass**: Use chain rule to compute the gradients of the loss with respect to each parameter



forward propagation
backpropagation
b_2
$\frac{\partial L}{\partial b_2} = \frac{\partial L}{\partial z_2}\frac{\partial z_2}{\partial b_2}$
$\frac{\partial z_2}{\partial b_2}$
h
$\frac{\partial z_2}{\partial h}$
z_2
y
$\frac{\partial L}{\partial h} = \frac{\partial L}{\partial z_2}\frac{\partial z_2}{\partial h}$
$\frac{\partial L}{\partial z_2}$
$\frac{\partial z_2}{\partial w_2}$
w_2
$\frac{\partial L}{\partial w_2} = \frac{\partial L}{\partial z_2}\frac{\partial z_2}{\partial w_2}$

3. **Parameter Update**: Apply gradient descent to minimize the loss

Types of Data

- **Images**
  - Represented as pixel matrices
  - 3D: [Channels, Height, Width]
- **Text**
  - Represented as sequences of tokens in a vocabulary
  - 1D: [Sequence Length]
- **Time Series**
  - Represented as sequences over Time
  - 1D: [Time]

Types of Models

- **Convolutional Neural Networks (CNNs)**
  - Mainly used for image data
  - Animation
- **Recurrent Neural Networks (RNNs)**
  - Designed for sequential data
- **Transformers**
  - Designed for handling long-range dependencies
  - Self-attention mechanism:

$$\text{Attention}(\mathbf{Q}, \mathbf{K}, \mathbf{V}) = \text{Softmax}\left(\frac{\mathbf{Q}\mathbf{K}^T}{\sqrt{d_{\mathbf{K}}}}\right)\mathbf{V} \tag{4}$$